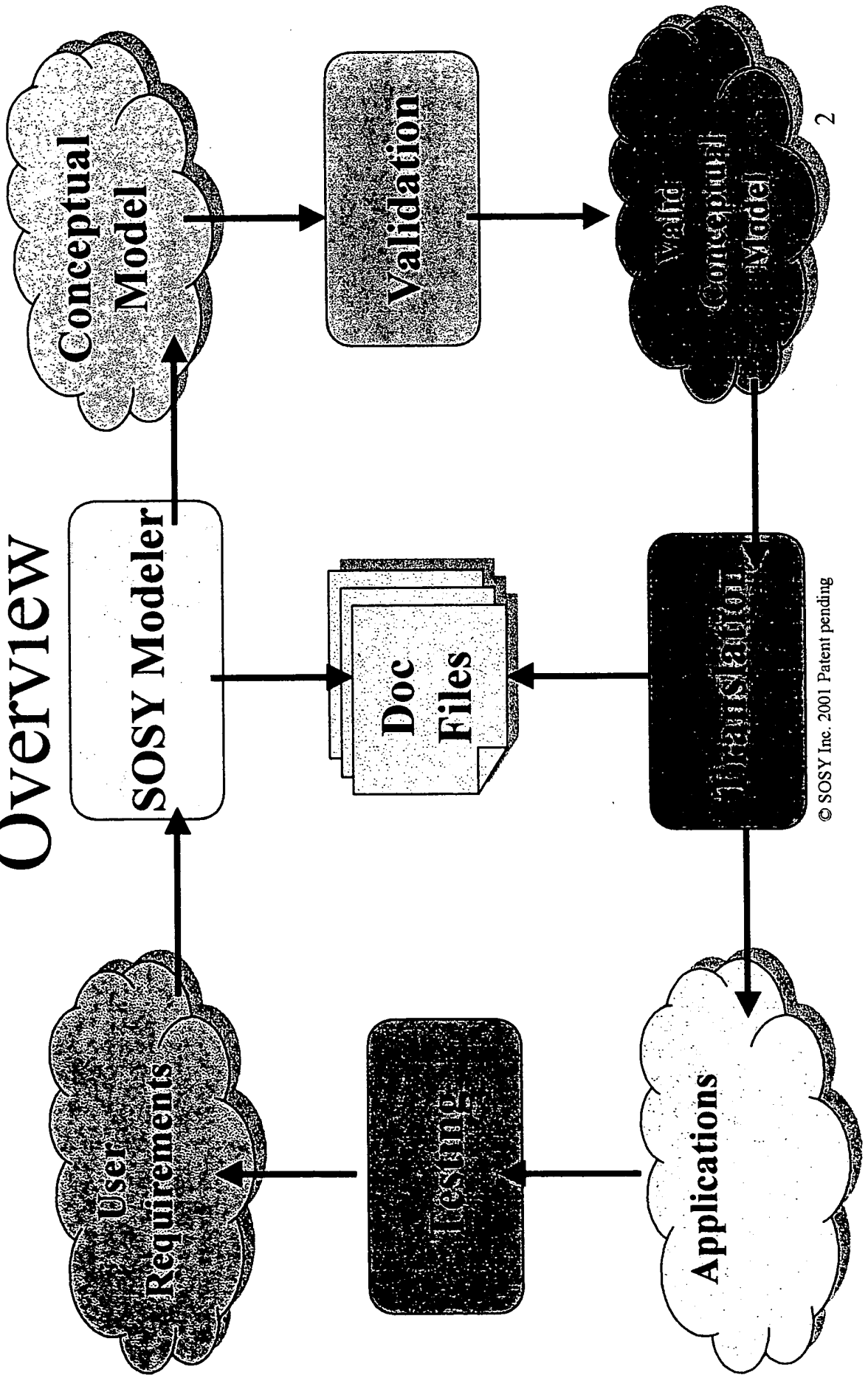


Summary

- Modelling
- Validation
- Documentation
- Persistence
- Business Logic
- User Interface

Overview



Conceptual Modeling Phase

CARE Technologies, S.A.

Index

- Intro
- Overview
- Phase 0. Requirements elicitation.
- Phase 1. Classes identification.
- Phase 2. Relationships between classes.
- Phase 3. Filling classes' details.

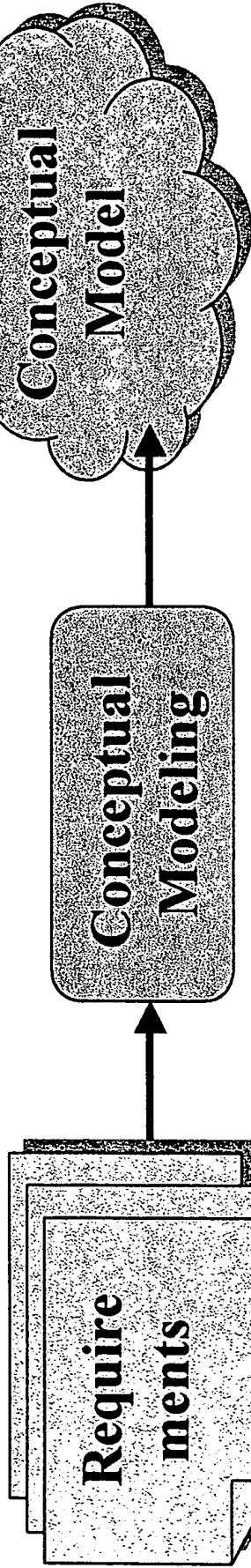
Index

- Phase 4. Express evaluations.
- Phase 5. Agent relationships.
- Phase 6. State Transition Diagram.
- Phase 7. Presentation Model.

Intro

- Conceptual Modeling Phase is a process of systematically & precisely description of the system to build, using:
 - Graphical UML compliant diagrams.
 - Constrains and semantics in a formal non-ambiguous language.
 - This phase is assisted by an integrated Modeler tool.

Overview



Requirements

- Specifications
- Documents
- Interviews
- Reports
- Other info. sources

Conceptual Model

- Classes
- Relationships
- Attributes
- Services
- ...

Expressed in a non-ambiguous language.

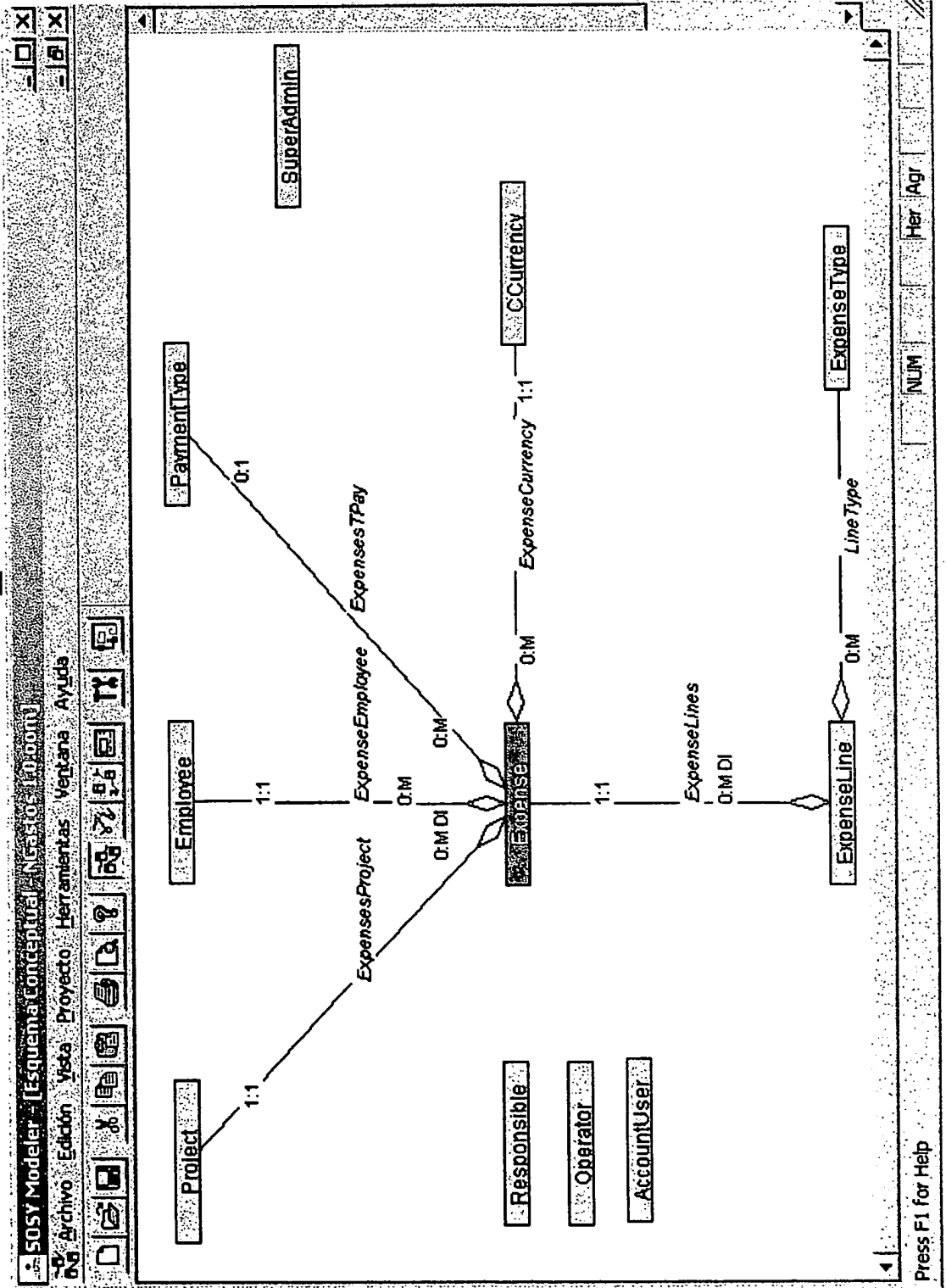
Phase 0. Requirement elicitation.

- Gathering the system requirements.
 - By meetings & interviews with customers, experts and final users.
 - By collecting reports, or documents expressing the system how-to and using tools.
 - Obtaining a coherent set of information as input to the next phase.

THE UNIVERSITY OF CHICAGO



Phase 2. Relationships between classes.



Phase 3. Filling classes' details.

Clase

Arbitratos

Servicios

Derivaciones

Restricciones

Agentes

Transacciones

Relaciones

Generalidades

Arbitratos

Nombre	Tipo atributo	Tipo dato	Id	Tamaño	Valor defecto	Pedir al crear	Nulos	Añadir
PresortDate	Constante	Date			today()	Si	No	Añadir
Status	Variable	Int			0	No	No	Modificar
Cause	Variable	String		255		Si	No	Borrar
AuthoDate	Variable	Date			NULL	No	Si	
AuthoComments	Variable	String		255	NULL	No	Si	
PaymentDate	Variable	Date			NULL	No	Si	
PayComments	Variable	String		255	NULL	No	Si	
TotExpenses	Derivado	Real						
TotExpensesCur	Derivado	Real						
Advances	Variable	Real			0	Si	No	
AdvancesCur	Derivado	Real						
Exchange	Variable	Real						
Balance	Derivado	Real				No	Si	
BalanceCur	Derivado	Real						

Limpiar

Notas >>

Nombre

Tipo Atributo

Tipo Dato

Alias

Observaciones

Clase:

Expense

Aceptar

Cancelar

[illegible]12

Phase 3. Filling classes' details.

Clase: **Expense** Acceptar Cancelar

Atributos | Servicios | Derivaciones | Restricciones | Agentes | Transacciones | Relaciones | Generalidades

Eventos y Transacciones

Servicios

Nombre	Características
newexpense	New
delexpense	Destroy
modify	
close	
authorize	
approve	
pay	
rejectautho	
rejectpayment	
insPaymentType	Shared with PaymentType
borPaymentType	Shared with PaymentType
DELETEALL	Tran
TPAY	Tran

Añadir | Modificar | Borrar | Copiar | Limpiar | Notas >>

Parámetros

Nombre	Tipo dato
p_thisExpense	Expense
p_Cause	String
p_Advances	Real
p_Exchange	Real

Añadir | Modificar | Borrar | Notas >>

Atributos

Nombre	Tipo dato
Nombre	
Tamaño	<input type="checkbox"/> Nulos <input type="checkbox"/> Colecciones
Alias	
Valor por defecto	
Observaciones	

Nombre:
 Alias:
 Mensaje de Ayuda:
 Observaciones:

Clase: **Expense** Acceptar Cancelar

Phase 3. Filling classes' details.

Clase: [Expense] [v]

Avituos | Servicios | Derivaciones | Restricciones | Agentes | Transacciones | Relaciones | Generalidades |

Transacción: [DELETEALL] [v]

Fórmula: [FOR ALL Lines DO Lines.deleteLine(Lines). deleteExpense(THIS)]

Acción: [Expense] [v]

Clase/Rol: [Expense] [v]

Agentes: [Servicio] [v] [approve] [v] [p_thisExpense] [v] [Crear Parámetro en la Transacción] [v]

Observaciones: [Observaciones]

Cancelar

Phase 3. Filling classes' details.

xl

Clase:

Atributos | Servicios | Derivaciones | Restricciones | Agentes | Transacciones | Relaciones | Generalidades

Estáticas

Exchange > 0

Añadir
Modificar
Borrar
Heredadas

Fórmula

Exchange > 0

Mensaje De Error

Exchange must be greater than zero

Dinámicas

Añadir
Modificar
Borrar

Fórmula

Oper. Temporal

Condición

Oper. Temporal

Condición2

Mensaje De Error

Clase:

Expense

Aceptar
Cancelar

Phase 4. Express evaluations.

Modelo Funcional

Clase: Expense Atributo: Cause

Evento: modify Efecto: p_Cause Condición:

☐ Cardinal ☒ De Estado ☐ De Situación

Detalles de Evaluación

Evento: modify

Condición de evaluación:

IF:

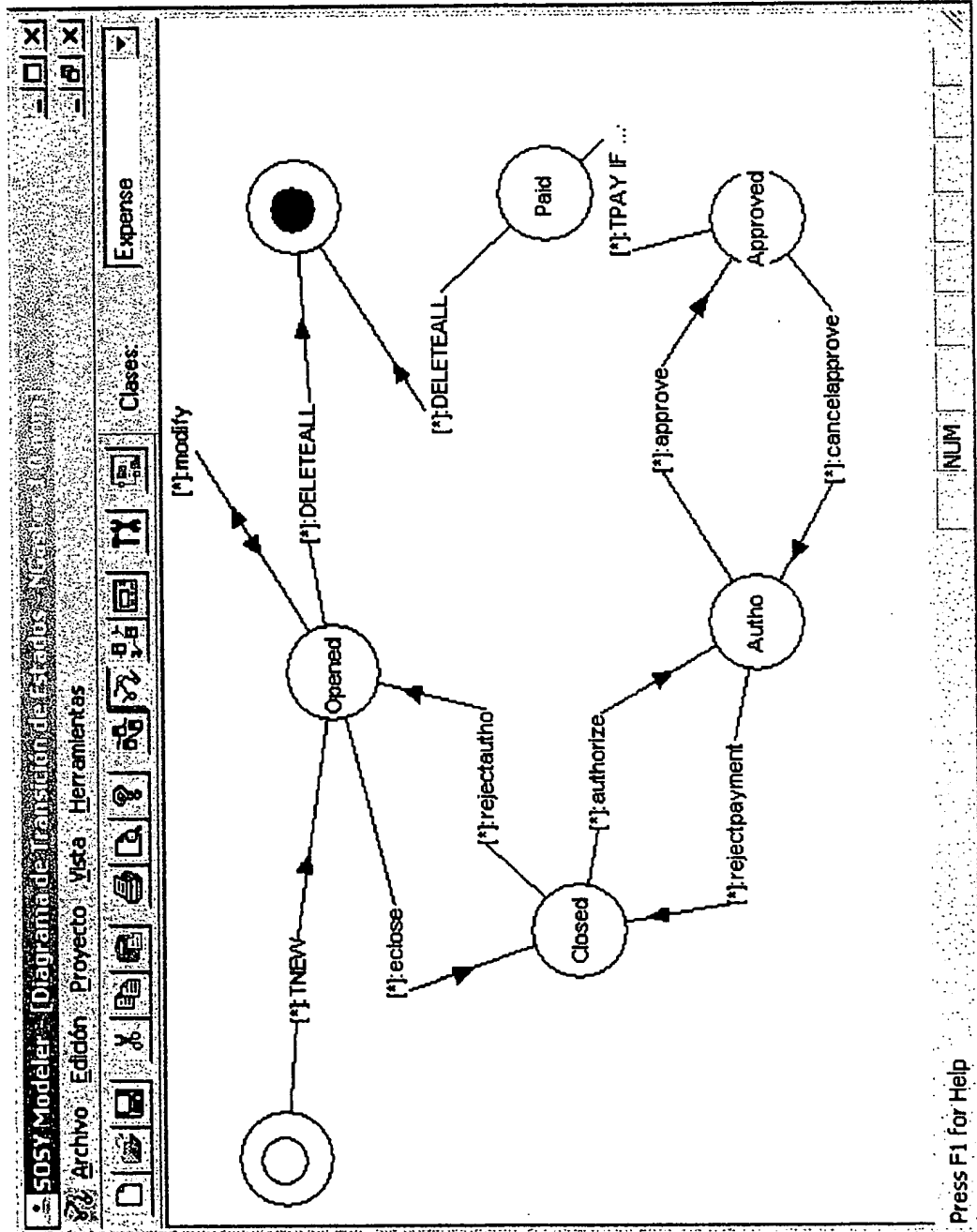
Efecto del evento:

p_Cause

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100																																																																																																																																																											
0	00000000	00000001	00000010	00000011	00000100	00000101	00000110	00000111	00001000	00001001	00001010	00001011	00001100	00001101	00001110	00001111	00010000	00010001	00010010	00010011	00010100	00010101	00010110	00010111	00011000	00011001	00011010	00011011	00011100	00011101	00011110	00011111	00100000	00100001	00100010	00100011	00100100	00100101	00100110	00100111	00101000	00101001	00101010	00101011	00101100	00101101	00101110	00101111	00110000	00110001	00110010	00110011	00110100	00110101	00110110	00110111	00111000	00111001	00111010	00111011	00111100	00111101	00111110	00111111	01000000	01000001	01000010	01000011	01000100	01000101	01000110	01000111	01001000	01001001	01001010	01001011	01001100	01001101	01001110	01001111	01010000	01010001	01010010	01010011	01010100	01010101	01010110	01010111	01011000	01011001	01011010	01011011	01011100	01011101	01011110	01011111	01100000	01100001	01100010	01100011	01100100	01100101	01100110	01100111	01101000	01101001	01101010	01101011	01101100	01101101	01101110	01101111	01110000	01110001	01110010	01110011	01110100	01110101	01110110	01110111	01111000	01111001	01111010	01111011	01111100	01111101	01111110	01111111	10000000	10000001	10000010	10000011	10000100	10000101	10000110	10000111	10001000	10001001	10001010	10001011	10001100	10001101	10001110	10001111	10010000	10010001	10010010	10010011	10010100	10010101	10010110	10010111	10011000	10011001	10011010	10011011	10011100	10011101	10011110	10011111	10100000	10100001	10100010	10100011	10100100	10100101	10100110	10100111	10101000	10101001	10101010	10101011	10101100	10101101	10101110	10101111	10110000	10110001	10110010	10110011	10110100	10110101	10110110	10110111	10111000	10111001	10111010	10111011	10111100	10111101	10111110	10111111	11000000	11000001	11000010	11000011	11000100	11000101	11000110	11000111	11001000	11001001	11001010	11001011	11001100	11001101	11001110	11001111	11010000	11010001	11010010	11010011	11010100	11010101	11010110	11010111	11011000	11011001	11011010	11011011	11011100	11011101	11011110	11011111	11100000	11100001	11100010	11100011	11100100	11100101	11100110	11100111	11101000	11101001	11101010	11101011	11101100	11101101	11101110	11101111	11110000	11110001	11110010	11110011	11110100	11110101	11110110	11110111	11111000	11111001	11111010	11111011	11111100	11111101	11111110	11111111

17

Phase 6. State Transition Diagram.



Phase 6. STD Preconditions

Transición	
Origen:	Destino:
Approved	Paid
<div style="text-align: right;"> <input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/> </div>	
<div> <div> Detalles </div> <div> <div> Agentes: <div>AccountUser</div> <div>SuperAdmin</div> </div> <div> Servicio: <div>TPAY</div> </div> </div> </div>	
Precondición: <div>Balance > 0 OR ps_ReturnAdvance = TRUE</div>	
Condición de control: <div></div>	
Mensaje en caso de Error: <div>Check the advanced money excess</div>	

Phase 7. Presentation Model.

Conjunto de Visualización

Nombre: CV_Expense
Limpia
Borra

Atributos a visualizar:

Atributo	Tipo dato
Project.ProjectName	String
Employee.EmpName	String
Employee.EmpSur...	String
Status	Int
AuthoDate	Date
PaymentDate	Date
TotExpenses	Real
Balance	Real

<< Añadir

Eliminar >>

Subir

Bajar

Agregar

Atributos:

Atributo	Tipo dato
Cause	String
AuthoDate	Date
AuthoComments	String
PaymentDate	Date
PayComments	String
TotExpenses	Real
TotExpensesCur	Real
Advances	Real
AdvancesCur	Real
Exchange	Real
Balance	Real
BalanceCur	Real

Clase: Expense

Aceptar
Cancelar

Phase 7. Presentation Model.

Filtro

flt_Expense

Alias:

Expense Reports

Limpiar

Borrar

Project = vf_Project AND Employee = vf_Employee AND PresentDate >= vf_DateIniIssue AND PresentDate <= vf_DateEndIssue AND AuthoDate >= vf_DateIniApp AND AuthoDate <= vf_DateEndApp AND PaymentDate >= vf_DateIniPay AND PaymentDate <= vf_DateEndPay AND

<< Variable

Observ

Variables

Nombre	Alias	Tipo dato	Tipo estilo	Estilo	Nueva	Modificar	Borrar
vf_Project	Project	Project	Sel. Población				
vf_Employee	Employee	Employee	Sel. Población				
vf_DateIniIssue	Initial Issuing Date	Date					
vf_DateEndIssue	Final Issuing Date	Date					
vf_DateIniApp	Initial Approving D	Date					

Tipo

Simple

Objeto-valuado

Nombre

Alias

Tipo de dato:

Estilo de introd.

Estilo de selección:

Clase: Expense

Aceptar

Cancelar

Conceptual Model Validation

CARE Technologies, S.A.

Index

- Intro
- Overview
- Validation Degrees
 - Partial Validation
 - Total Validation

Index

- Validation Types
 - Elements of the Conceptual Model
 - Formulas of the Conceptual Model (Syntax)
- Validation Trees
 - Nodes
 - Leaves
- Example

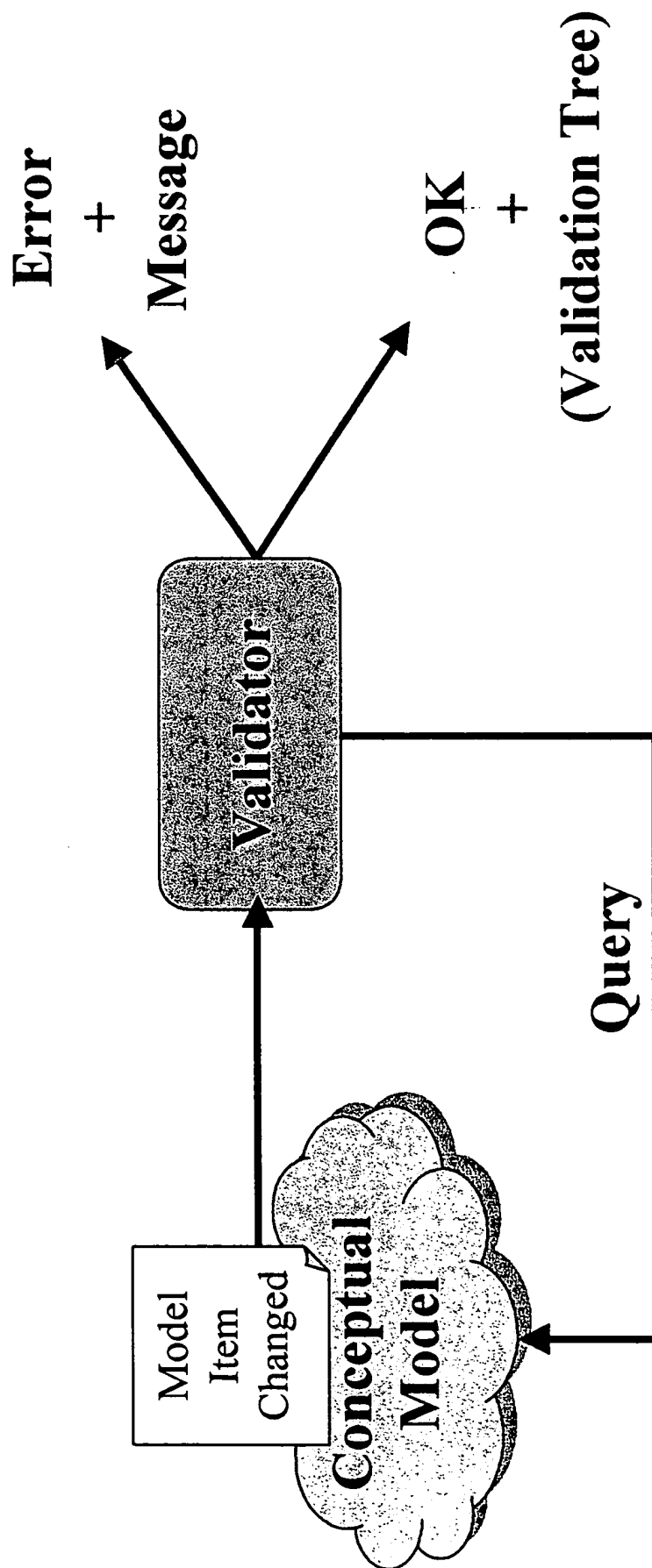
Intro

- Conceptual Model Validation is the process by which a conceptual model or a modification of it is proven to be valid:
 - Correct
 - Non Ambiguous
 - Non Contradictory
 - Complete
 - Every concept is fully specified
- Validation process checks the representation of requirements in Formal Specification Language to be valid

Validation Degrees

- Partial Validation
 - That of a single element of the Conceptual Model.
 - Happens whenever an element is added, modified or deleted.

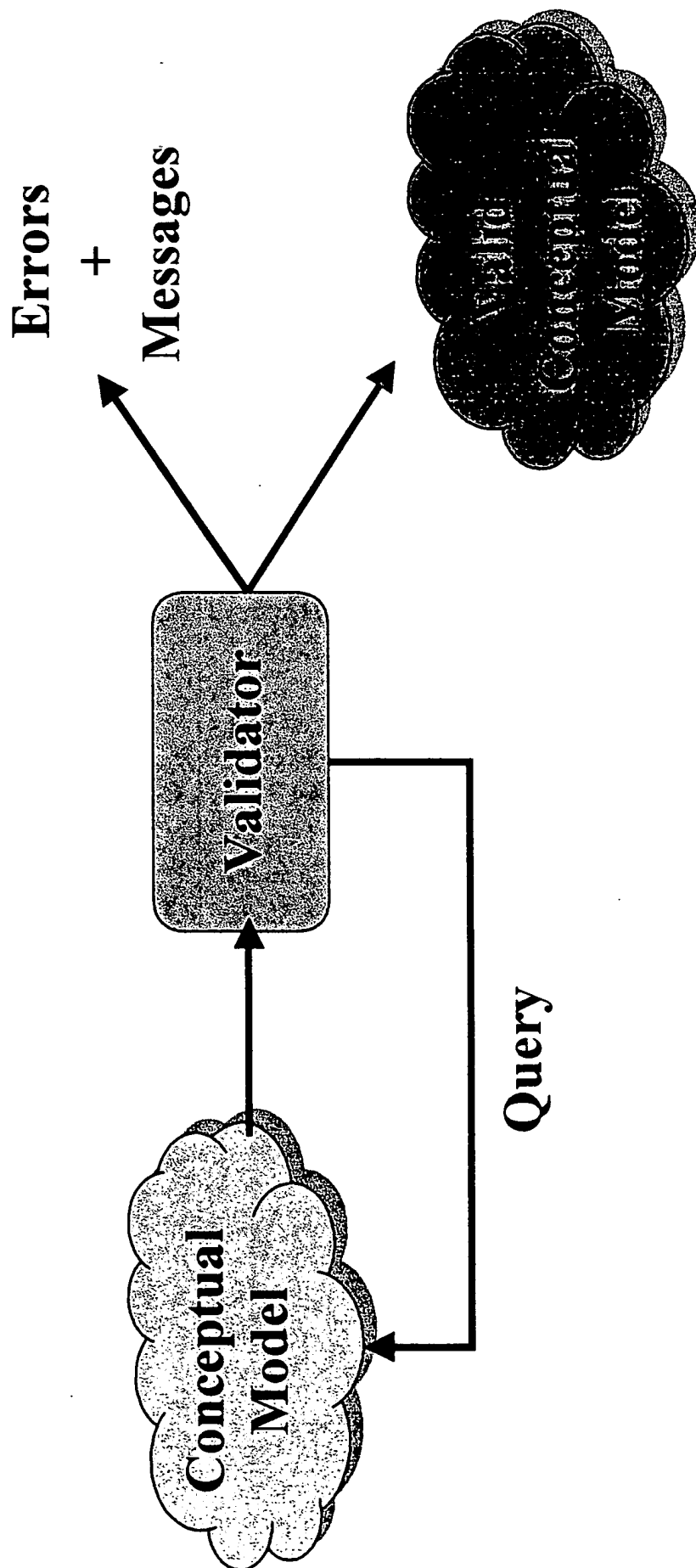
Partial Validation Overview



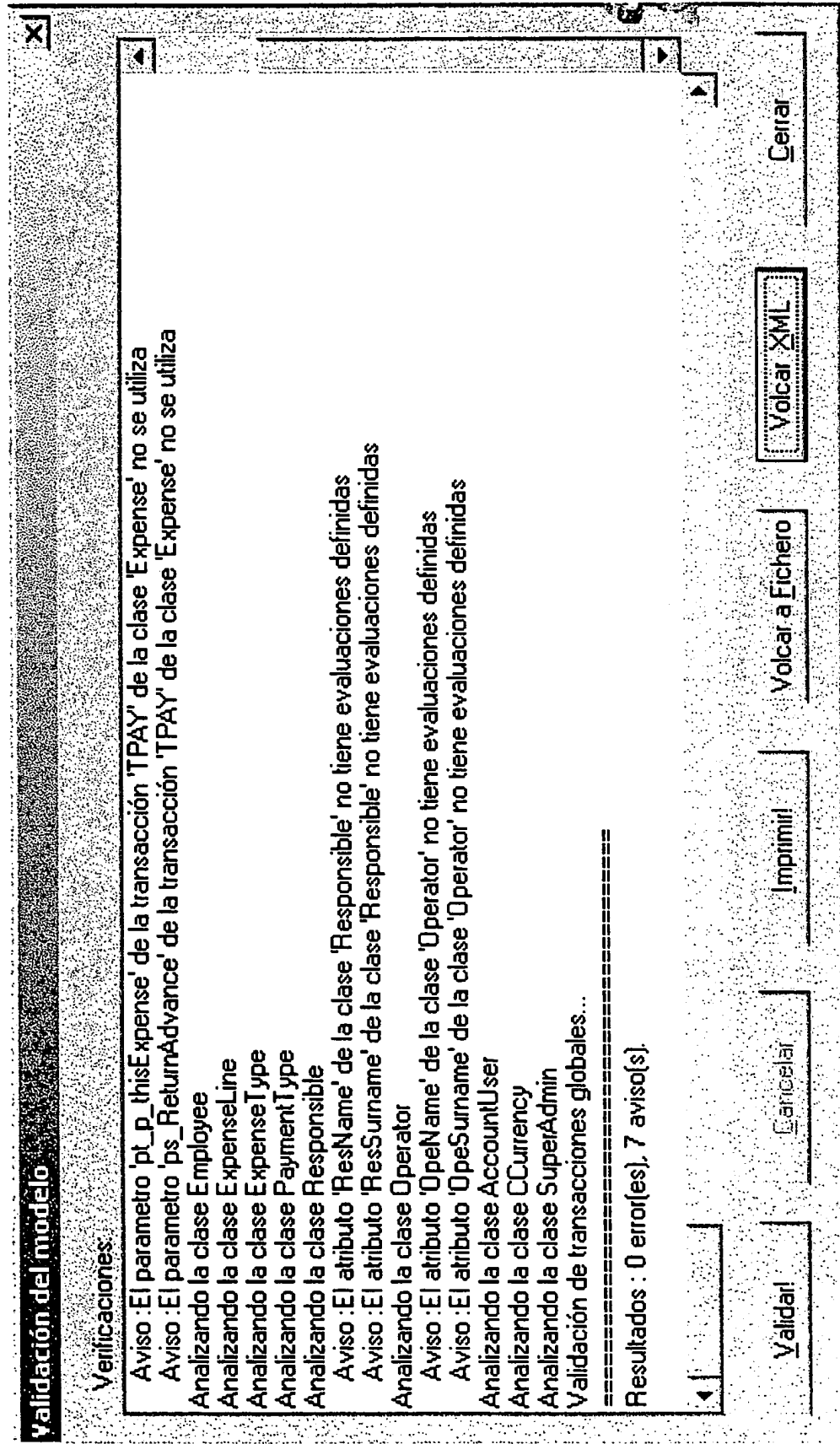
Validation Degrees

- Total Validation
 - That of the whole Conceptual Model.
 - Happens by request.
 - Must happen prior to any translation process.
 - Takes advantage of partial validations already performed.

Total Validation Overview



Total Validation Example



Validation Types

- Elements of the Conceptual Model
 - Ensure the properties of an element (except formulas) are correct and complete.
 - Conditions that must hold depend on the type of element and the property being validated.
 - Examples:
 - Class Name is unique in a Conceptual Model.
 - Attribute Name is unique in its Class (but not in a Conceptual Model)

Validation Types

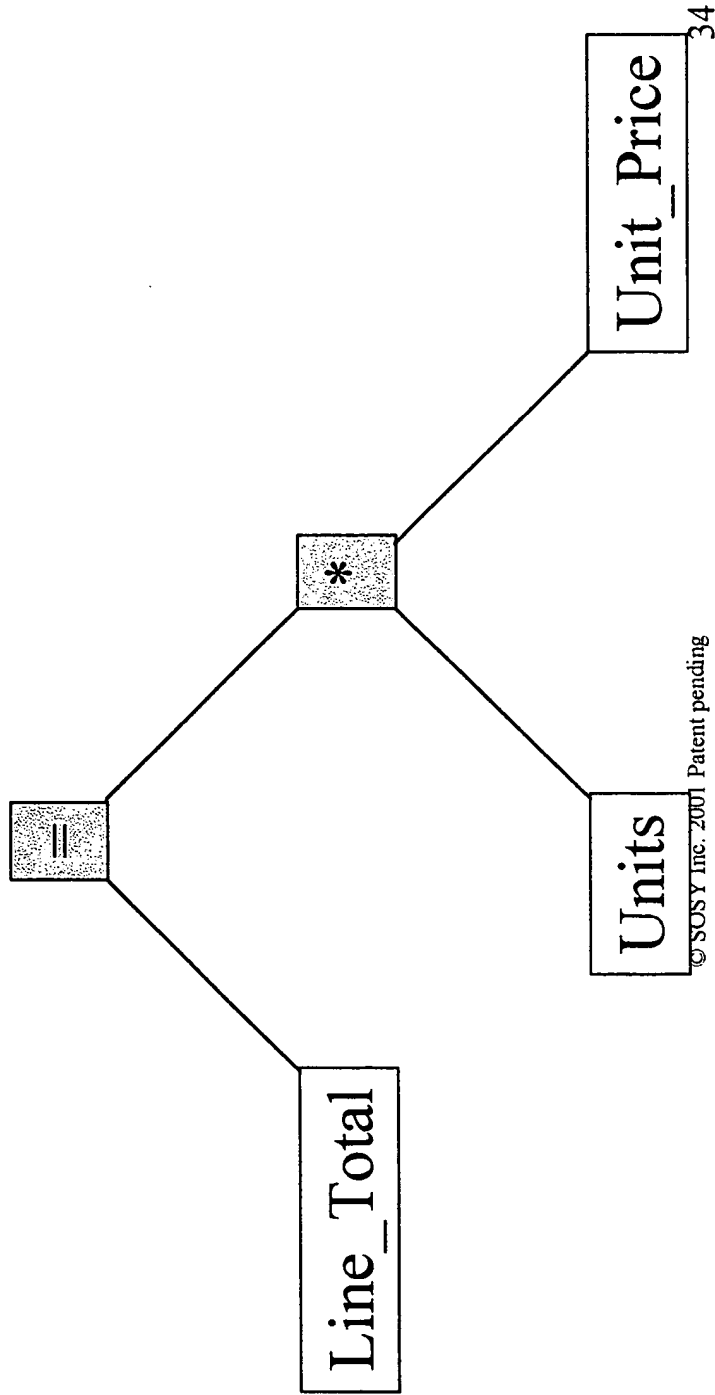
- Formulas of the Conceptual Model
 - Ensure the formulas of the Conceptual Model are correct and complete.
 - Syntactical and Semantical Validation according to an extended Formal Specification Language grammar.
- Input:
 - Formula expression
 - Formula Type (precondition, valuation, ...etc.)
 - Formula Context (class name, service name, ...etc.)
- Output:
 - Error Message (validation did not pass)
 - Validation Tree (validation passed)

Validation Trees

- Binary Tree representation of a correct formula.
- Tree consists of Nodes and Leaves.
- Nodes
 - Represent operators
 - Can have one or two “branches” (binary)
 - Branches can again be nodes or leaves
- Leaves
 - Represent operands
 - Have no branches

Example

- $\text{Line_Total} = \text{Units} * \text{Unit_Price}$



Documentation Translation

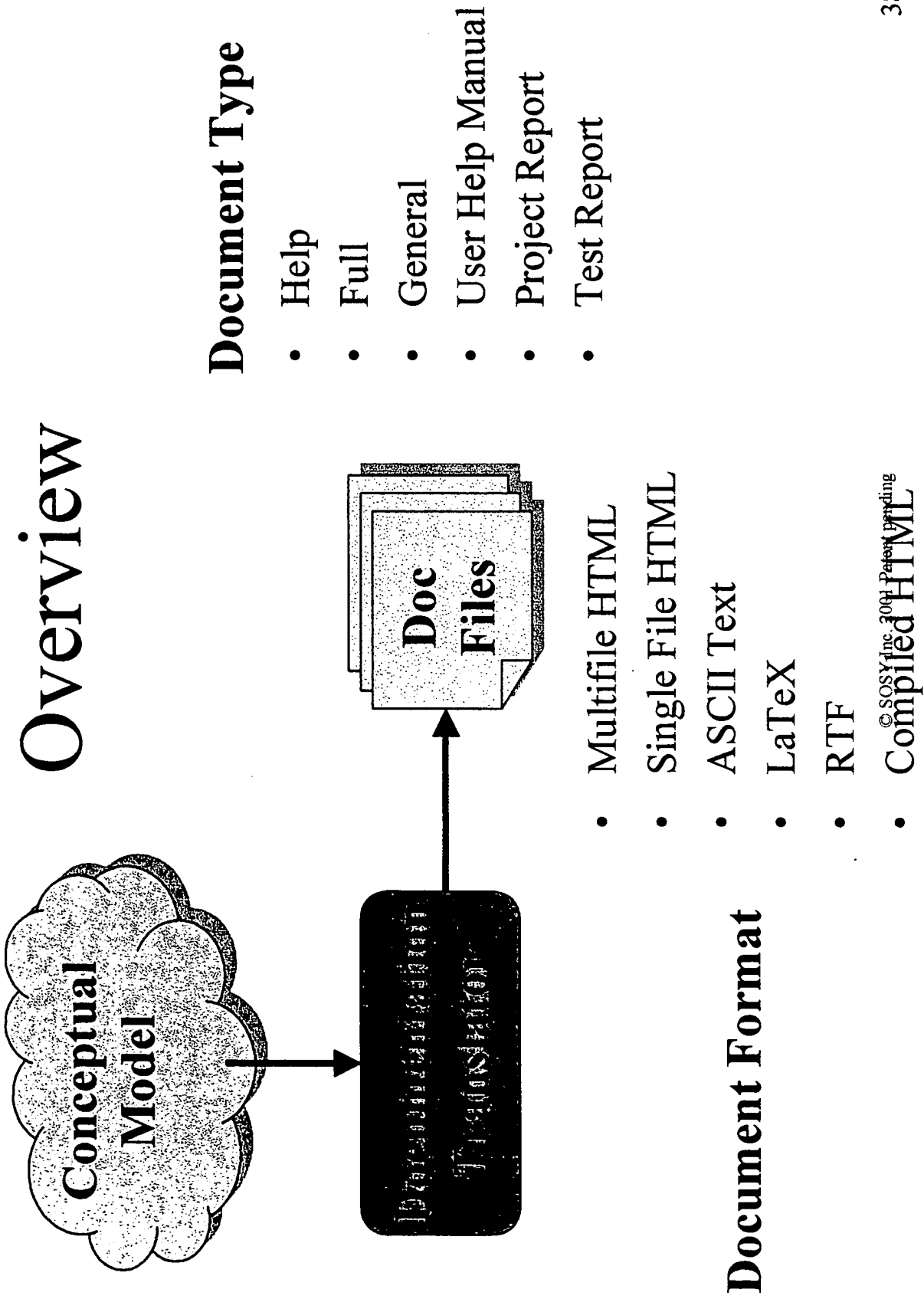
CARE Technologies, S.A.

Index

- Intro
- Overview
- Output Detail
 - Document Types
 - Document Formats
- Translation
 - CM Subset of Interest
 - Translation Process
 - Remarks
- Example

Intro

- Documentation Translation is the process to obtain, from a Conceptual Model, documentation on the system it represents.
- Documentation can have several degrees of detail and be focused on different aspects, thus obtaining different documentation formats from the same Conceptual Model.



Output Detail

- Document Types
 - Help
 - Description of each Class, its Attributes, Services and Population Selection Filters.
 - Full
 - Full description of a Conceptual Model
 - Aimed at analysts.
 - General
 - Description of each Class Attributes, Identification Function, Services, Aggregation Relationships and Specialization Relationships.

Output Detail

- Document Types
 - User Help Manual
 - Both Help Manual and Contextual Help (F1 key).
 - Intended for Operation Manual.
 - Integration with User Interface applications.
 - Project Report
 - Description of each Class Attributes and Services.
 - Test Report
 - Description of each Class Services.
 - Intended for Testing purposes.

Output Detail

- Document Formats
 - Multifile HTML
 - One HTML page per concept.
 - Recommended for navigable help.
 - Single File HTML
 - One single HTML page.
 - Recommended for printing.
 - ASCII Text
 - Single, plain ASCII text file.

Output Detail

- Document Formats
 - LaTeX
 - Single, LaTeX text file.
 - RTF
 - Single, RTF text file.
 - Compiled HTML
 - Same as Multifile HTML plus header files to be used by HTML Help Workshop compiler.
 - Recommended for contextual help.
 - Searching and Indexing facilities usage from browsers.

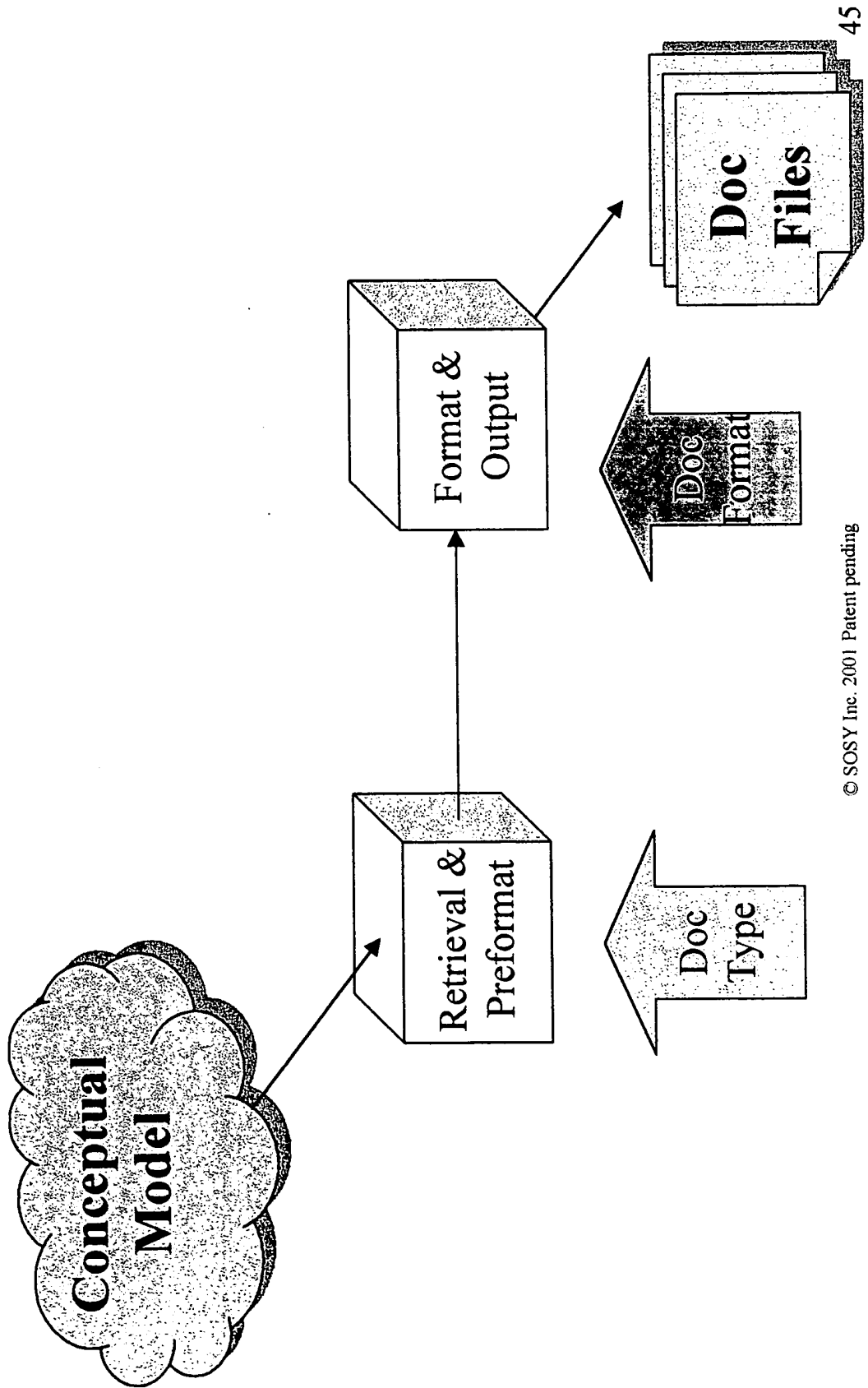
Translation

- Conceptual Model Subset of Interest
 - Subset of Interest depends on Document Type.
 - Usual elements:
 - Classes
 - Attributes
 - Relationships
 - Services & Arguments
 - Intensive use of analysis information.

Translation

- Translation Process
 - Read information from Conceptual Model and format it for output.
 - Two phases:
 - Information retrieval and pre-formatting.
 - Depends on Document Type
 - Independent from Document Format
 - Information output.
 - Depends on Document Format.
 - Independent from Document Type.

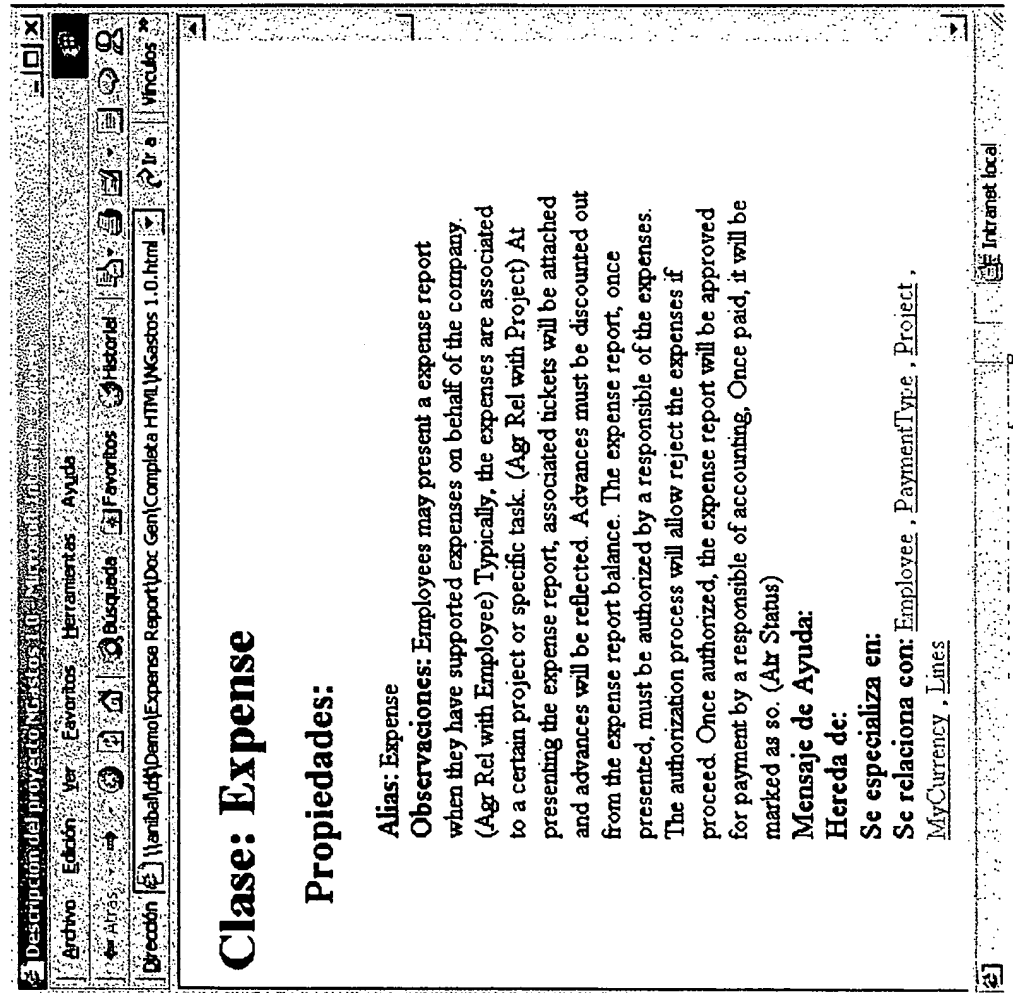
Translation Phases



Translation

- Remarks
 - Conceptual Model needs not to be valid (in terms of completeness and correctness) but it is always non-ambiguous.
 - The richer the analysis information, the richer the documentation.
 - Easily extensible
 - New Document Types
 - New Document Formats

Example



Persistence Relational Database Translation

CARE Technologies, S.A.

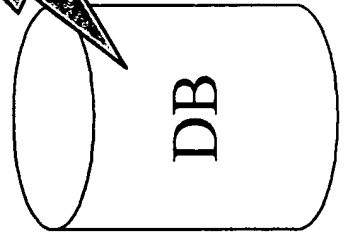
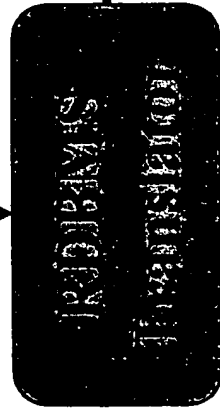
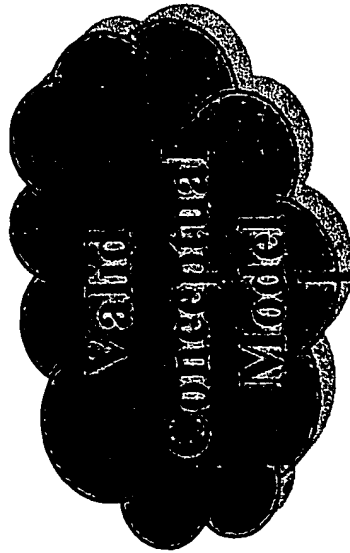
Index

- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Persistence Relational Database Translation is the process of creating a Relational Database from a certain subset of information in the Object Model of a valid Conceptual Model.
- Output script files are used to create a relational database using structured query language (SQL).

Overview



- Creates
- Primary Keys
- Foreign Keys
- Indexes
- Drop Creates
- Drop Primary Keys
- Drop Foreign Keys
- Drop Indexes

Output Detail

- **Creates**
 - Creation of Tables and Fields
- **Primary Keys**
 - Creation of Primary Keys as constraints on each table
- **Foreign Keys**
 - Creation of Foreign Keys as constraints on each table
- **Indexes**
 - Creation of Indexed on each table

Output Detail

- Drop Creates
 - Deletion of Tables
- Drop Primary Keys
 - Deletion of Primary Key Constraints
- Drop Foreign Keys
 - Deletion of Foreign Key Constraints
- Drop Indexes
 - Deletion of Indexes

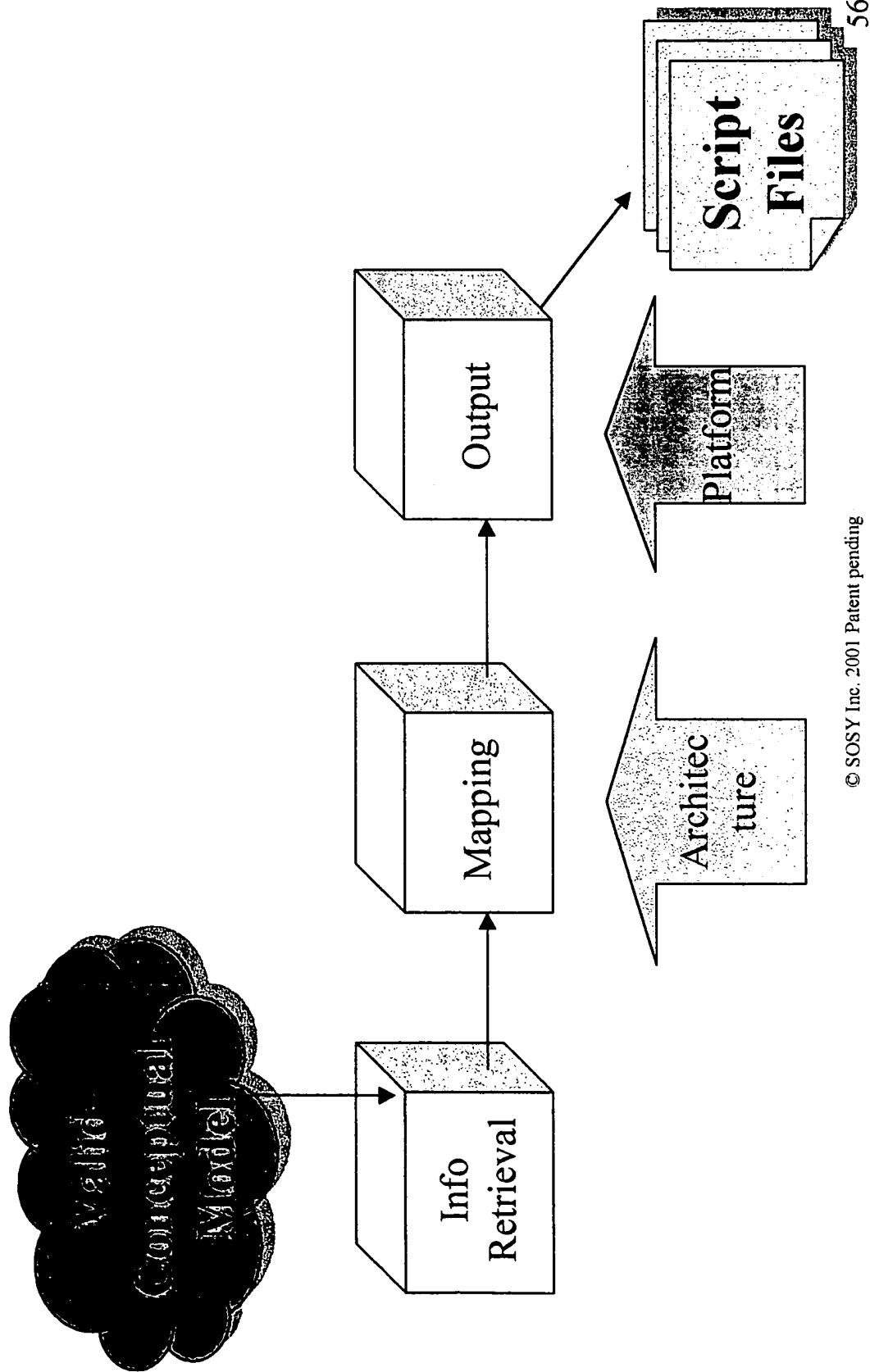
Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Classes
 - Attributes
 - Identification Functions
 - Aggregation Relationships
 - Inheritance Relationships

Translation

- Three phases:
 - Information retrieval.
 - Independent from persistence architecture.
 - Fixed architecture mapping.
 - Depends on persistence architecture.
 - Information output.
 - Targeted for Standard ANSI SQL 92 RDBMS.
 - Script files depends on the platform's SQL syntax of RDBMS manufacturer.
 - May depend on platform specifications to make use of manufacturer extensions and tuning.

Translation Phases



Translation

- Translation Processes. Mapping:
 - Class \rightarrow Table
 - Non-derived Attribute \rightarrow Field
 - Identification Function \rightarrow Primary Key
 - Univaluated Relationship \rightarrow Foreign Key
 - Univaluated Relationship \rightarrow Index
 - Multivaluated Relationship \rightarrow Table
 - Inheritance Relationship \rightarrow Foreign Key

Example

Create table script in SQL for Expense class

```
CREATE TABLE Expense (
    fk_Project_1 int NOT NULL ,
    id_Expense int NOT NULL ,
    fk_Employee_1 CHAR(10) NOT NULL ,
    fk_MyCurrency_1 CHAR(5) NOT NULL ,
    fk_PaymentType_1 CHAR(5) NULL ,
    PresentDate datetime NOT NULL ,
    Status int NOT NULL ,
    Cause VARCHAR(255) NOT NULL ,
    AuthoDate datetime NULL ,
    AuthoComments VARCHAR(255) NULL ,
    PaymentDate datetime NULL ,
    PayComments VARCHAR(255) NULL ,
    Advances DECIMAL(19,6) NOT NULL ,
    Exchange DECIMAL(19,6) NOT NULL);
```

Business Logic Translation

CARE Technologies, S.A.

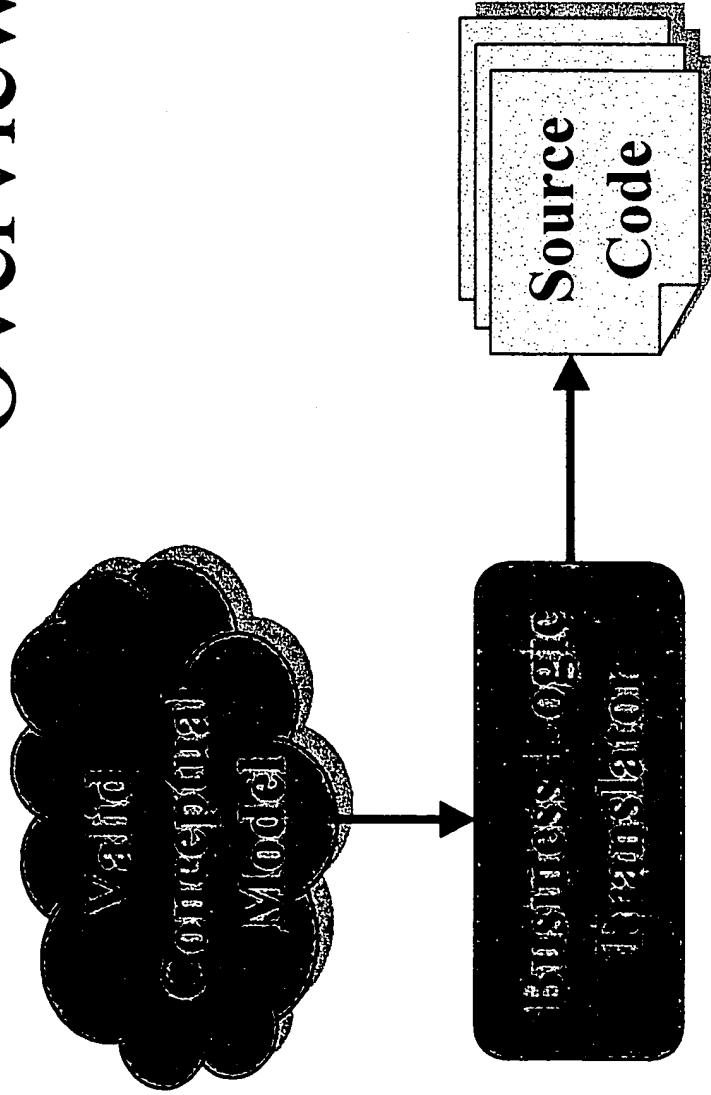
Index

- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Business Logic Translation is the process to obtain, following a precise Execution Model, the source code corresponding to the business logic from a valid Conceptual Model for a target Programming Language and Software Architecture.
- Execution Model is independent from Programming Language and Software Architecture.

Overview



Determines:

-Target Programming Language

-Target Software Architecture

Output Detail

- Target Programming Language and Software Architecture determine:
 - Source code organization in files
 - Files internal organization
- Source Code's backbone: Execution Model.

Output Detail

- Traceability: Source code highly readable and maintainable thanks to:
 - Source code is always organized and structured in the same way.
 - Naming conventions applied.
 - Source code includes analysis information from the Conceptual Model as comments.

Output Detail

- Implementation of a precise Execution Model grants Functional Equivalence with Conceptual Model.
- Programming Interface to Clients for:
 - Actor Validation and Authentication.
 - Services Execution.
 - Queries Execution.
- Manages:
 - Concurrency.
 - Transactions.
 - Interoperable Object Persistence.

Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Static properties (Visibility & Persistence)
 - Attributes + Identification Functions
 - Derivations
 - Aggregation Relationships
 - Inheritance Relationships
 - Services (Execution Model)
 - Arguments
 - Preconditions
 - Transaction Formulas
 - Actors (Execution Model)
 - Integrity Constraints (Execution Model)

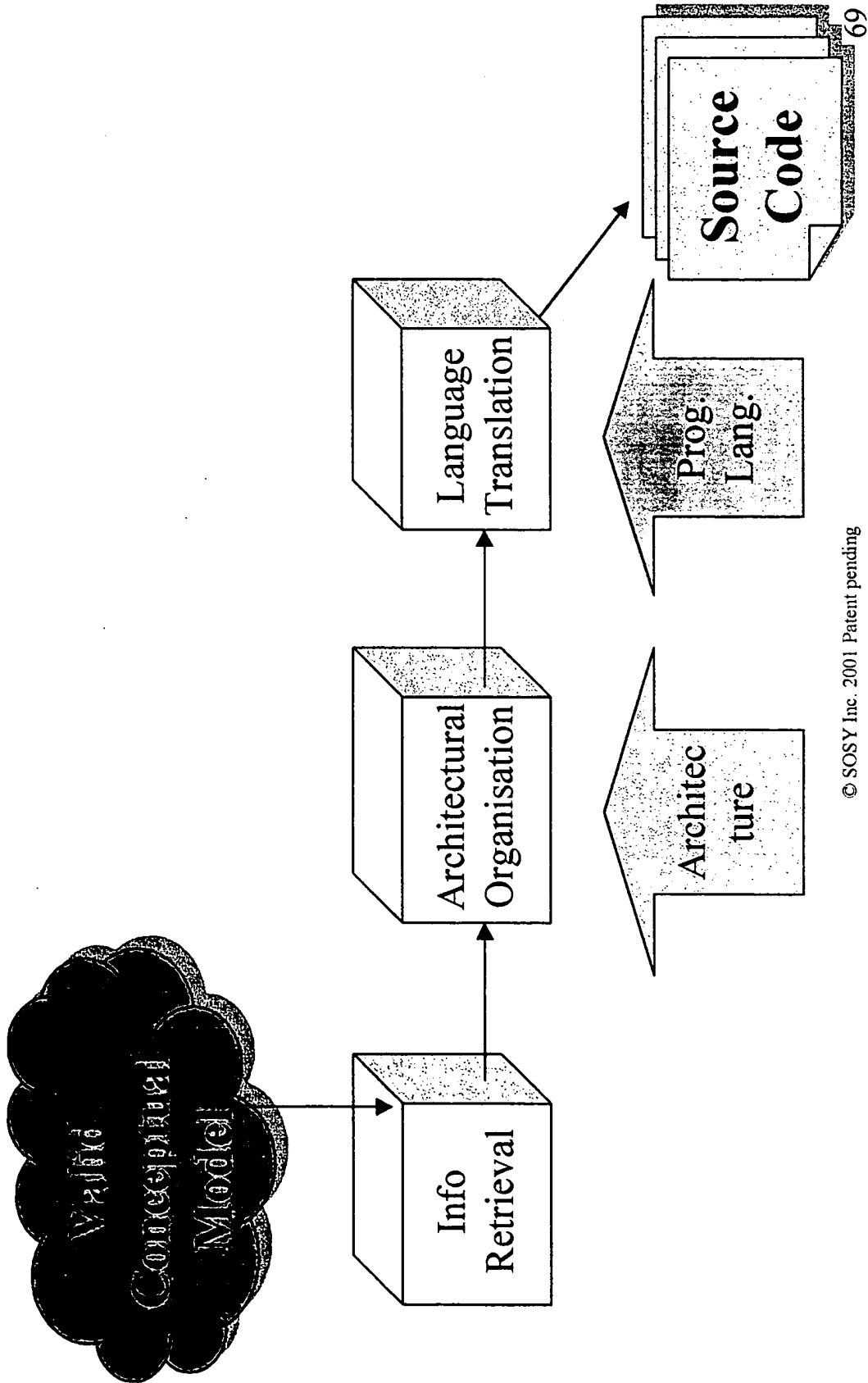
Translation

- Conceptual Model Subset of Interest.
 - Dynamic Model.
 - State Transition Diagram (Execution Model).
 - Controls Valid Lives for an Object.
 - Object Interaction Diagram.
 - Triggers (Execution Model).
 - Global Transactions (Execution Model).
- Functional Model (Execution Model).
 - Object state change upon occurrence of an event.

Translation

- Translation phases:
 - Information retrieval
 - Independent from target Software Architecture and Programming Language
 - Architectural organisation
 - Depends on target Software Architecture
 - Independent from target Programming Language
 - Determines files organisation and files internal structure
 - Language translation
 - Depends on target Programming Language
 - Influenced by Software Architecture
 - Takes advantage of Programming Language capabilities

Translation Phases



Translation

- Translation Processes
 - Classes
 - Static properties translation
 - Services translation
 - Queries translation
 - Global Interactions
 - Services translation
 - Global Functions
 - Functions Interface translation
 - Body is left blank

Example

- Evaluation:
 - Service Authorize modifies attributes Status, AuthoDate and AuthoComments
 - Formal Specification Language expression for evaluation Valuation
- [authorize ()] Status=2 and AuthoDate=today() and AuthoComments="";
- Visual Basic Produced

```
Private Function MV_Eval_Expense_authorize() As String
    Expense_Status = 2
    Expense_AuthoDate = today()
    Expense_AuthoComments = ""
    MV_Eval_Expense_authorize = ""
End Function
```

User Interface Translation

CARE Technologies, S.A.